Abstract—Inverted pendulums have been used for a long time as a typical laboratory setup for learning and testing of various control system techniques. In this paper, we present a new laboratory rotary inverted pendulum setup. The setup is equipped with a microcontroller containing a code for communicating directly with MATLAB/Simulink using Simulink Desktop Real Time. In this configuration students have the opportunity to implement a controller in MATLAB/Simulink or to implement it directly on the microcontroller. Its capabilities are tested using sliding mode control, first using simulation and then verified on the experimental setup.

Keywords—inverted pendulum, sliding mode control, Furuta pendulum

I. INTRODUCTION

The rotary inverted pendulum is a commonly used benchmark in control systems laboratories for learning and testing of various control system techniques. It consists of an arm driven by a motor which is able to rotate in the horizontal plane and a pendulum suspended from the arm which is able to rotate in the vertical plane. It is also called Furuta pendulum. It has one input and two degrees of freedom and belongs to a class of nonlinear unstable underactuated mechanical systems [1], [2].

For Furuta pendulum a large number of papers are written presenting different linear and nonlinear controllers for swing-up and stabilizing the pendulum in the upright unstable position.

In [3], [4] Linear Quadratic Regulator (LQR) is used for stabilizing the pendulum in the upright position, while additional swing-up controller is used for swinging up the pendulum. In presence of nonlinearities, such as dead-zone caused by static friction, the performance of the controller degrades and a limit cycle can occur. For such case, the limit cycle can be avoided by designing a linear controller in the frequency domain taking into account nonlinearity described by describing function [5]. A design based on differential flatness and root locus is used for selecting state feedback gains in [6].

Besides linear control techniques, various nonlinear techniques are also reported in the literature [7]. Model predictive control is used for swinging up and/or stabilizing the pendulum in upright position [8]–[10]. Proportional retarded controller is reported successful in stabilizing the pendulum in case of time delays [11].

Among nonlinear control techniques the sliding-mode control is a popular nonlinear control technique due to robustness against system uncertainties and external disturbances. However, standard sliding-mode control is not directly applicable for underactuated systems without selecting an appropriate sliding surface [12]. Sliding mode controllers are reported for both swing-up and stabilization of Furuta pendulum in [12]–[14], proving semiglobal asymptotic stability. Furthermore sliding mode controllers are often used for comparison with other controllers such as active disturbance rejection control combined with differential flatness [15].

The simulation model can be written in terms of differential equations representing dynamics of the system, which can be obtained by using Euler-Lagrange or Newton-Euler methods [16]–[18], while some authors resort to making a 3D model in CAD software such as and exporting the simulation model directly into SimMechanics [18]–[20].

Several papers presented building their own inverted pendulum systems (both rotary and pendulum on a cart) [19], [21], [22], concentrating on the mechanical design, design of the simulation model and experiments. Papers use either a commercial rapid prototyping system, such as dSPACE DS1104, [19] or an inexpensive microcontroller, such as Arduino, to implement the controller [22].

In this paper an experimental model of Furuta pendulum built for experimental verification of various control algorithms taught in undergraduate, graduate and postgraduate control courses, such as linear state feedback, model predictive control or sliding-mode control is presented. The built experimental setup is equipped with a microcontroller with prepared code for communicating directly with MATLAB/Simulink using Simulink Desktop Real Time. The microcontroller controls two H-bridges with current sense for driving up to two DC motors and it is able to provide either motor voltage or control the motor current (torque) directly. In the first phase, students use the designed interface to run the control algorithms directly from Simulink using Simulink Desktop Real Time, while in a later phase they can exploit added JTAG connector on the PCB to implement the control algorithm directly on the microcontroller. Sliding mode control is selected to show the possibility for testing a high bandwidth nonlinear control algorithm directly from MATLAB/Simulink.

The rest of the paper is organized as follows. In Section II the mathematical model of the pendulum is presented. Section III presents the experimental setup. Section IV presents the designed control algorithm and Section V presents simulation and experimental results. Section VI concludes the paper.
II. PENDULUM MODEL

The Furuta pendulum, shown in Fig 1, consists of an arm which is able to rotate in the horizontal plane and a pendulum suspended from the arm which is able to rotate in the vertical plane. The pendulum angle is denoted by \( \theta \) and arm angle is denoted by \( \alpha \), respectively, while their meaning is shown in Fig 1. The arm is driven directly by an electric motor.

![Fig. 1. Furuta pendulum model](image)

Dynamic equations of the system can be obtained using Euler-Lagrange or Newton-Euler methods. The full nonlinear system dynamics for the Furuta pendulum with a complete inertia tensor is reported in [16]. This model can be simplified for simulation purposes by considering only the total moment of inertia of the pendulum arm and the pendulum about their pivot points: \( J_r = J_r + m_p l_p^2 \) and \( J_p = J_p + 0.25m_p l_p^2 \), respectively. In those expressions, \( J_r \) represents the moment of inertia of the arm around z axis, \( J_p \) moment of inertia of the pendulum around its center of mass, \( m_p \) pendulum mass, \( l_r \) length of the arm and \( l_p \) length of the pendulum.

The equations are [16]:

\[
\begin{align*}
\left( \dot{J}_r + J_p \sin^2 \alpha \right) \ddot{\theta} - \frac{1}{2} m_p l_p l_r \cos \alpha \ddot{\alpha} + \\
J_p \sin(2\alpha) \dot{\theta} \dot{\alpha} + \frac{1}{2} m_p l_p l_r \sin \alpha \dot{\alpha}^2 &= \tau - B_r \dot{\alpha}, \tag{1}
\end{align*}
\]

\[
\begin{align*}
-\frac{1}{2} m_p l_p l_r \cos \alpha \ddot{\theta} + \dot{J}_p \ddot{\alpha} - \frac{1}{2} J_p \sin(2\alpha) \dot{\theta} + \frac{1}{2} m_p l_p g \sin \alpha &= -B_p \dot{\alpha}. \tag{2}
\end{align*}
\]

\( B_r \) is DC motor and \( B_p \) pendulum friction coefficient. The torque \( \tau \) is generated by the DC motor and can be expressed as

\[
\tau = \frac{n_c m (u - n_c \dot{\theta})}{R_a}. \tag{3}
\]

In the upper equation, \( n \) is the gearbox ratio, \( R_a \) armature resistance, \( c_e \) and \( c_m \) are back-EMF and torque motor constants.

III. EXPERIMENTAL SETUP

A custom experimental setup shown in Fig 2 has been designed. It consist of a DC motor with gearbox equipped with a magnetic encoder for driving the arm and measuring its position and 1024 ppr incremental encoder for measuring the position of the pendulum.

A custom made PCB is designed, which consist of a microcontroller, dual H-bridge with current sense, USB port, connectors for connecting 3 incremental encoders, and 2 motors. A custom code is written to communicate with MATLAB S-function and a Simulink block diagram has been made. However, the PCB is equipped with a JTAG connector to enable students downloading their own custom code. Furthermore, H-bridge with current sense enables for two different modes of operation. The control input can be either the DC motor voltage or armature current (motor torque). The latter, enables students to use equations of motion directly obtained from Euler-Lagrange equations.

![Fig. 2. The experimental setup](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m_r )</td>
<td>( 5.3 \times 10^{-3} )</td>
<td>kg</td>
</tr>
<tr>
<td>( l_r )</td>
<td>0.09</td>
<td>m</td>
</tr>
<tr>
<td>( J_r )</td>
<td>( 4.6 \times 10^{-3} )</td>
<td>kg \cdot m^2</td>
</tr>
<tr>
<td>( m_p )</td>
<td>0.107</td>
<td>kg</td>
</tr>
<tr>
<td>( l_p )</td>
<td>0.3395</td>
<td>m</td>
</tr>
<tr>
<td>( J_p )</td>
<td>( 3.5 \times 10^{-6} )</td>
<td>kg \cdot m^2</td>
</tr>
<tr>
<td>( B_r )</td>
<td>0.056</td>
<td>kg \cdot m^2/s</td>
</tr>
<tr>
<td>( B_p )</td>
<td>( 2.5 \times 10^{-4} )</td>
<td>kg \cdot m^2/s</td>
</tr>
<tr>
<td>( g )</td>
<td>9.81</td>
<td>m/s^2</td>
</tr>
<tr>
<td>( c_m )</td>
<td>0.0134</td>
<td>Nm/A</td>
</tr>
<tr>
<td>( c_e )</td>
<td>0.0134</td>
<td>Nm/rad</td>
</tr>
<tr>
<td>( R_a )</td>
<td>3.33</td>
<td>( \Omega )</td>
</tr>
<tr>
<td>( n )</td>
<td>54.7945</td>
<td></td>
</tr>
</tbody>
</table>

TABLE I
PARAMETERS OF THE ROTARY PENDULUM SIMULATION MODEL
IV. Controller Design

For controller synthesis, the following simplified model is used:

\[
\begin{align*}
\ddot{\theta} &= -a_m \dot{\theta} + c_e u, \\
\dot{\alpha} &= -\frac{C}{J_p} \dot{\theta} + \frac{m_p g l_p}{2 J_p} \sin \alpha + \frac{K}{J_p} \dot{\theta},
\end{align*}
\]

where \( J_p = J_p + 0.25m_p l_p^2 \). The upper equation represents simplified model of the permanent magnet DC motor which is driving the rotating base, where the constant \( a_m \) includes back-EMF and friction effects and \( c_e \) effect of the input voltage on the angular acceleration of the base. The bottom equation describes the dynamics of the pendulum. \( C \) is the pendulum friction, \( m_p \) and \( J_p \) pendulum mass and moment of inertia and \( l_p \) pendulum length. \( K \) is a proportionality constant which is \( K > 0 \) for inverted and \( K < 0 \) for non-inverted position. The only control input is the DC motor voltage \( u \) [23].

The system (4) can be rewritten as

\[
\begin{align*}
\ddot{\theta} &= -a_m \dot{\theta} + c_e u, \\
\ddot{\alpha} &= -\frac{C}{J_p} \dot{\alpha} + \frac{m_p g l_p}{2 J_p} \sin \alpha + \frac{K}{J_p} \dot{\theta},
\end{align*}
\]

The next step is to transform the system (5) into the new form using [23]

\[
\begin{align*}
y &= \theta - \frac{J_p}{K} \alpha, \\
x &= \dot{y} - \frac{C}{K} \alpha,
\end{align*}
\]

which results in the following system formulation:

\[
\begin{align*}
x &= -\frac{m_p g l_p}{2K} \sin \alpha, \\
\dot{y} &= x + \frac{C}{K} \alpha.
\end{align*}
\]

### A. Sliding Mode Controller

Let’s take the state component \( \theta_1 \) in the upper equation of the system (7) as control input. If the last term of this equation satisfies

\[
\sin \alpha = -\lambda_1 (x + y)
\]

and

\[
\lambda_1 = -\frac{2K}{m_p g l_p}, \quad \frac{m_p g l_p}{2C} > \frac{\pi}{2}, \quad K > 0,
\]

then the system (7) is asymptotically stable. Consequently, the system (5) is asymptotically stable. For more information regarding system stability, refer to [23].

Equation (8) holds if the function

\[
s_1 = \sin \alpha + \lambda_1 (x + y) = 0.
\]

The condition \( s_1 = 0 \) will be satisfied, if the function \( s_1 \) satisfies the linear first order differential equation

\[
\dot{s}_1 = -\lambda s_1, \quad \lambda > 0.
\]

The expression (11) will be satisfied if the sliding mode is enforced in the switching surface

\[
s = s_1 + \lambda s_1 = \cos \alpha \dot{\alpha} + \lambda_1 (\dot{x} + \dot{y}) + \lambda s_1 = 0.
\]

The time derivative of \( s \) equals

\[
\dot{s} = \frac{K c_e}{J_p} \cos \alpha u + \Psi(x, y, \alpha, \dot{\alpha}),
\]

where \( \Psi \) is a nonlinear function of the system states. For the controller to achieve a stable behaviour, the functions \( s \) and \( \dot{s} \) need to have opposite signs. This is achieved if

\[
\dot{s} = -\eta \text{sign}(s), \quad \eta > 0,
\]

in which case the control input equals

\[
u = -\frac{J_p}{K c_e \cos \alpha} (\Psi + \eta \text{sign}(s)).
\]

However, such formulation can cause unwanted chattering. In order to reduce this effect, (14) was replaced by

\[
\dot{s} = -\eta \tanh(s/2), \quad \eta > 0.
\]
In the following chapters, control law realized with (14) will be referred to as "Controller 1" and the one realized with (16) as "Controller 2".

B. Swing-Up Controller

To bring the pendulum in the upright position, we use energy control strategy described in [24]. The energy of the pendulum is

\[ E = \frac{1}{2} \dot{\theta}^2 + \frac{1}{2} m g l \cos \theta, \]  

(17)

while the reference energy equals maximum potential energy

\[ E_0 = m g l. \]  

(18)

There are various control laws which can be used to achieve reference energy (18). The one we use in this work keeps the amplitude of control signal constant, while changing only its sign, and is given by

\[ u = -\mu \text{sign}(E - E_0) \alpha \cos \alpha. \]  

(19)

Swing-up controller is active until the condition \(|\alpha| < 30^\circ\) is satisfied, when it switches to sliding mode control law (15).

V. SIMULATION AND EXPERIMENTAL RESULTS

Verification of described algorithms has been done using both simulation and experimental tests. Controller parameters are \( \lambda = 10 \), \( \eta = 40 \) and swing-up gain \( \mu = 12 \). At the beginning of the experiment, pendulum is in its stable stationary position with all the system states \([\theta \ \dot{\theta} \ \alpha \ \dot{\alpha}]^T = [0 \ 0 \ 0 \ 0]^T\).

Fig. 4 shows simulation results for both controller versions. In the upper graph, it can be seen that Controller 1 yields faster \( \theta \) angle dynamics, but is unable to hold the base of the pendulum in the zero position, whereas Controller 2 causes oscillations and results in zero steady state tracking error. Pendulum angle \( \alpha \) dynamics is shown in the lower graph and no difference in dynamics between the two controllers can be noticed. Both stabilize pendulum very fast and with no steady state error.

Input signals from the simulation can be seen in the Fig. 5. Swing-up phase lasts for cca. 1 second, after which the sliding mode is turned on. There, one can notice already mentioned chattering effect, which causes input signal to have extremely fast switching behaviour and causes high jerk when applied to an electromechanical system. Controller 2 doesn’t cause chattering, since around \( s = 0 \) it behaves similarly to a proportional controller and no switching is enforced.

Fig. 6 shows measured angles \( \theta \) and \( \alpha \). It can be seen that, in reality, neither of controllers is able to hold zero steady state \( \theta \) error. Controller 1 makes the system oscillate slightly around \( \theta \approx 100^\circ \), while Controller 2 results in a better behaviour and shifts those oscillations around \( \theta = 0^\circ \). This behaviour is consistent with observations made based on simulation, but with higher steady state error because of modelling errors and mechanical disturbances (e.g. friction, gearbox backlash etc.). On the other hand, both controllers show almost identical behaviour when it comes to \( \alpha \) angle control. Behaviour is very similar to the one obtained from the simulation.

Input signals in Fig. 7 show that swing-up time in experiment lasted similarly long as in the simulation. After the sliding mode control has been activated, Controller 1 behaves very similar to the simulated one, causing a lot of chattering. However, in reality Controller 2 also causes chattering, although smaller than Controller 1.

VI. CONCLUSION

In this paper a new rotary inverted pendulum experimental setup has been proposed. The proposed experimental setup has a custom built PCB board equipped with a microcontroller with a code for communicating directly with MATLAB/Simulink using Simulink Desktop Real Time. In this configuration students have opportunity to implement controller in MATLAB/Simulink or implement the controller directly on the microcontroller. To show the capability of implementing high bandwith control algorithms, a sliding mode controller has been synthesised and implemented in MATLAB/Simulink. The controller has been tested in simulations and experimentally for two different versions of the controller. The first version uses sign function in the computation of the control signal while the second version uses \( \tanh \) for reducing the chattering
problem. In both simulation and experimental results the chattering has been reduced by using tanh function. The results on the proposed experimental setup closely resemble simulation results, which shows that the experimental setup can be used for testing various advanced nonlinear control algorithms.

ACKNOWLEDGMENT

This research has been supported by the European Regional Development Fund under the grant KK.01.1.01.0009 (DAT-ACROSS).

REFERENCES


